Open Set Recognition for Time Series Classification

Tolga Akar[2(✉)], Thorben Werner[1], Vijaya Krishna Yalavarthi[1], and Lars Schmidt-Thieme[1]

[1] Information Systems and Machine Learning Lab, University of Hildesheim, Universitatsplätz 1, 31141 Hildesheim, Germany
{werner,yalavarthi,schmidt-thieme}@ismll.de

[2] Distributed Artificial Intelligence Laboratory, Technical University of Berlin, Straße des 17. Juni 135, 10623 Berlin, Germany
tolga.akar@tu-berlin.de

Abstract. Traditional classification problems often assume that the number of classes present in the data is finite. This may hold true for the training data, but in real life, the risk of encountering unknown samples is ubiquitous. Classifying these unknown samples into one of the target classes can have drastic effects in some situations like security systems or body sensors. To address this problem, recently, open set recognition models that can correctly classify the known samples and detect the unknowns simultaneously, are proposed. In contrast to the existing models where unknown detection depends on the classification model, we propose, to the best of our knowledge, an open set recognition model for time series classification that works independent of the classifier by employing class-specific barycenters. Specifically, DTW distance, and the cross-correlation between the class-specific barycenters, and the input are used for detecting the unknown classes during testing. Our extensive experimental evaluation on the UEA multivariate time series archive with 30 datasets shows that the proposed open set recognition architecture deployed on top of the InceptionTime outperforms the state-of-the-art open set recognition models by an average of 22% in terms of macro F1 score.

Keywords: Open set recognition · Time series classification · Machine learning

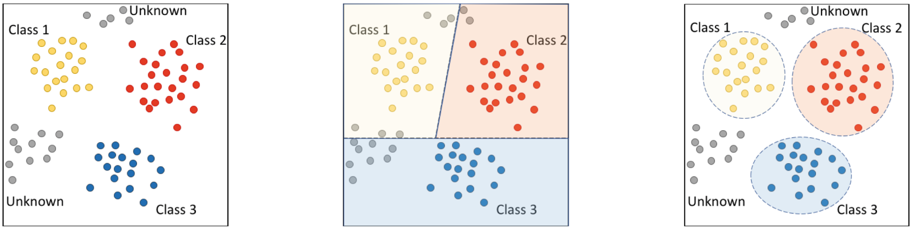
1 Introduction

The success of machine learning based solutions for various classification problems is undeniable. Most of the time, the number of target classes is assumed to be finite, and solutions for these problems are derived in such a way. However, in real-life applications, there is always a risk of encountering samples from

T. Akar—This work is done while the main author was doing his master's thesis at ISMLL, University of Hildesheim.

J. Gama et al. (Eds.): PAKDD 2022, LNAI 13281, pp. 354–366, 2022.
https://doi.org/10.1007/978-3-031-05936-0_28

unknown classes that are not seen during the training. This will, inevitably, lead to a situation where the classifier will classify those unknown samples as one of the target classes, which is of course wrong (e.g. Fig. 1b). Such wrong predictions can have drastic effects in certain situations, e.g. security systems, body sensors, machinery maintenance. To address this problem, open set recognition (OSR) models that can correctly classify the known samples and detect the unknowns at the same time are proposed in the past decade, starting with [15]. Even though OSR has received a lot of attention in recent years, the majority of the studies in this field focuses on computer vision problems, and best of our knowledge, there is no other work for open set recognition that focuses on the time series classification (TSC) task.



(a) (b) (c)

Fig. 1. Comparison between traditional classification (Fig. 1b) and open set recognition (Fig. 1c). Figure 1a shows the distribution of the original dataset.

This study focuses on proposing a methodology for achieving a solution for the open set recognition problem regarding the TSC tasks, that is generally applicable to multiple datasets and that can ideally work with different classifiers. The proposed method uses class-specific time series barycenters, i.e. the centroids representing a cluster of time series, for unknown detection. The DTW distance and the cross-correlation between a class-specific barycenter and an input determine whether the input belongs to the given class or not. As for the classifier, the proposed method benefits the state-of-the-art time series classification model InceptionTime [7]. Thus, the proposed method is referred to as Open Set InceptionTime (OS-InceptionTime or OS-IT). As the unknown detection methodology is independent of the classifier, it can be used alongside any other time series classification algorithm.

The scientific contribution of this study thesis is threefold:

- The first-ever open set recognition method that is specifically designed for time series is introduced. This answers the research question: *How can the open set problem be solved for the time series classification tasks?*
- It is shown that artificially created unknown samples can simulate the actual unknowns to some extent, eliminating the need for handpicking and thus, fully automating the training process. This answers the research question: *Can the training procedure be fully automated for the proposed method?*

– The proposed model is evaluated on 30 datasets to demonstrate that it is a generally applicable solution for open set problems regarding the various type of TSC datasets and can be used with any classifier. This answers the research question: *Is the proposed method generic enough to be applied for different types of TSC datasets?*

2 Background and Problem Definition

Let f_c be a traditional closed set classifier that takes a time series sequence x with L number of time steps and M number of dimensions (also referred as channels or variables) and assigns this input x a label y, deciding among the K number of known classes, i.e. $f_c : \mathbb{R}^{L\times M} \rightarrow \{1, ..., K\}$. An open set model f_o, on the other hand, has an extra possible class $K + 1$ to assign for the unknown samples, $f_o : \mathbb{R}^{L\times M} \rightarrow \{1, ..., K, K + 1\}$. The data that are seen during training that belong to the one of the known classes are referred to as the known samples and they are denoted with $\mathcal{D}_k$. A subset of $\mathcal{D}_k$ is used for training.

In most of the open set approaches, some sort of data that can represent the unknowns are either needed to train the model, or more commonly, to optimize the unknown detection thresholds after the training. These types of unknown samples that are used during the training will be referred to as the known unknowns. They are denoted with $\mathcal{D}_a$, having a vector of K+1 values as their labels. The last type of samples is the unknowns that are not a part of the training. They are referred to as the unknown unknowns. They may or may not appear during the testing in real-life applications. They are denoted with $\mathcal{D}_u$. Since all the samples that do not belong to the $\mathcal{D}_k$ are treated as unknowns, $\mathcal{D}_a$ can be considered as a subset of the $\mathcal{D}_u$ as well. In short, an ideal open set classifier f for an input x should predict the correct class label k for known samples, and $K + 1$ for the unknowns as follows:

$$f(x) = \hat{y} = \begin{cases} k & \text{for } x \in \mathcal{D}_k,\ k \in \{1, ..., K\} \\ K+1 & \text{for } x \in \mathcal{D}_u \end{cases} \tag{1}$$

3 Related Work

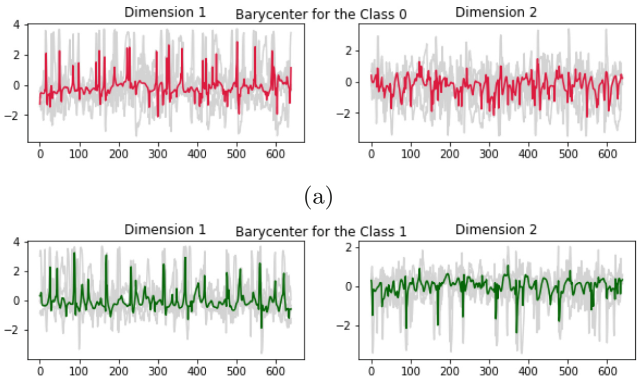
Barycenters. There are multiple ways to compute barycenters. The first one, Euclidean, is simply the arithmetic mean of each point in time. It is much faster to compute than the others, however, it does not provide a meaningful representation enough since it does not take shifts in time into account like the DTW (Dynamic time warping) based methods. The second one, originally proposed in [13], is an iterative averaging method to compute the barycenters under DTW. The aim is to minimize the DTW distance between the center (average sequence) and the actual sequences in the given class or dataset. Expectation-maximization or stochastic subgradient methods are used to find optimal solutions with this method. Unlike the DBA (DTW Barycenter Averaging) approach, soft-DTW, introduced in [3], uses a differentiable loss function to solve this minimization problem, which makes it much more easier to obtain the optimal result. In other words, the soft-DTW method is able to find more accurate and smoother barycenters for sequential data. Thus, for this study, barycenters are computed using the soft-DTW geometry. An example result of this calculation can be seen in Fig. 2.

Time Series Classification. After their proven success [10], convolutional neural networks (CNNs) attracted a great amount of attention from the TSC community who were looking for scalable alternatives to traditional ensemble classifiers such as HIVE-COTE [11] or BOSS [14]. In [6], authors experimented on several CNN based deep learning solutions for TSC and reported that Fully Connected Neural Networks (FCNs) and deep Residual Networks (ResNet) achieved the best performances overall. More recently, following the footsteps of [6], InceptionTime method is proposed in [7] and shown to achieve state-of-the-art performance, on par with HIVE-COTE.

Open Set Recognition. In the last decade, the popularity of the open set recognition (OSR) domain has grown significantly after [15] revealed that unknown samples can generate high activation scores for some of the known classes in closed set classifiers. The first application of OSR on deep networks was [2], where authors introduced a novel OpenMax layer. During testing, this OpenMax layer replaces the final softmax layer, which enables the classifier to have a probability distribution with an extra class probability for the unknown class. Directly extending the OpenMax paper, [8] proposes G-OpenMax, which utilizes a generative adversarial network (GANs) to generate samples of unknown classes. The trend of using generative models for OSR tasks continued with Class Conditioned Auto-Encoder for Open-set Recognition (C2AE) [12] and Classification-Reconstruction Learning for Open-Set Recognition (CROSR) [18], both using slightly different auto-encoder networks, and are similar in the way that both of them uses EVT to decide on the reconstruction thresholds. CGDL [16] can be considered the state-of-the-art OSR method with a generative network. It uses a variational auto-encoder (VAE) which is forced to approximate different Gaussian models for different known classes. Another alternative approach, [5] introduces two novel loss functions for unknown detection, that maximize the entropy of the unknown samples, namely Entropic Open-Set Loss and Objectosphere Loss. The only OSR paper for time series is [9], however, they are using a specific dataset for combustion engines rather than one of the popular TSC benchmark datasets, and they perform open set recognition to label each time step, but not the time series as a whole sequence.

4 Methodology

The unknown detector of the proposed model consists of two separate criteria to minimize the chance of missing unknown data. The first one is the DTW



(a)

(b)

Fig. 2. Barycenters for the AtrialFibrillation dataset for the classes 0 and 1. The samples that belong to that class are in the background in gray.

similarity. It is basically the sum of squared distances between a barycenter and a sample, computed after aligning both time series using DTW. If the distance between a sample and the barycenter is above a certain threshold for all the known classes, that sample is considered unknown. In cases where the intraclass variation is high, barycenters are usually not able to represent meaningful patterns regarding that class. In such cases, an out-of-class sample that looks like a horizontal line along the mean can have a smaller distance, especially in low dimensional time series, than a sample that actually belongs to that class. For this reason, a second criterion is added to the unknown detector.

The second criterion is the cross-correlation (a.k.a sliding dot product) of a sample and a barycenter. Similar to the convolution operation, cross-correlation is mainly used for searching an input sequence for a given filter (usually a shorter filter representing a feature). In this case, the barycenter functions as the filter and slid through the input sample to calculate the cross-correlation. The idea behind this approach is that out-of-class samples should generate much lower cross-correlation values. Cross-correlations are computed for each dimension of the data separately. If a sample generates a lower cross-correlation value for at least one of the dimensions for a specific class, then, it is rejected by that class for extra safety.

Compared in isolation, the cross-correlation criterion works usually better than the distance criterion. However, there are some datasets where the cross-correlation threshold does not work well. Hence, combining the two yields better results in most of the datasets used in this study. The formulas for defining the thresholds are quite straightforward. In Eqs. 2 and 3, τ_k^{dist} and τ_k^{cc} represent the distance and the cross-correlation thresholds for a known class k. For each k, median distance to the barycenter of the class $\tilde{\mu}_k^{dist}$, median maximum cross-correlation $\tilde{\mu}_k^{cc}$ with the barycenter, and their standard deviations σ_k^{dist} and σ_k^{cc} are computed using the train samples belonging to the class k. Median values are chosen here over the means in order to reduce the effect of the outliers within the class.

$$\tau_k^{dist} = \tilde{\mu}_k^{dist} + \alpha \cdot \sigma_k^{dist}, \quad for\ k \in \{1, ..., K\} \tag{2}$$

$$\tau_k^{cc} = \tilde{\mu}_k^{cc} - \beta \cdot \sigma_k^{cc}, \quad for\ k \in \{1, ..., K\} \tag{3}$$

The crucial values in these equations are hyper-parameters α and β, since they determine the magnitude of the thresholds. A grid search among the combinations of possible values (ranging from 0 to 5) is performed using the whole train set in order to find the optimal values. The full outline of the inference procedure can be seen in Algorithm 1.

Since this study aims to propose a generic solution that is applicable to multiple datasets, the actual unknown samples are not used in this stage to prevent cherry-picking. Instead, an artificial set of unknown data (known unknowns) are generated for each dataset (see Algorithm 2) and used to evaluate the open set performance of the model. The aim of the grid search is to find optimal hyper-parameters that can help detect the unknowns while maintaining high accuracy for the known samples. To do this a simple formula (Eq. 4) is used to assess the performance after every iteration. Then, the combination of hyper-parameters with the highest score $s(\alpha, \beta)$ is chosen.

$$s(\alpha, \beta) = \lambda^4 \cdot \frac{acc_X \cdot acc_A}{acc_X + acc_A} \tag{4}$$

$$\lambda = \frac{acc_X}{acc_{closed\ set}} \tag{5}$$

Given a train set X and artificially created known unknown data matrix A, acc_X stands for the accuracy of the model with the given hyper-parameters for the original train samples (known classes) X. Similarly, acc_A is the accuracy for detecting the known unknown samples, i.e. the recall for the $(K+1)$th (the label for unknowns) class. The λ functions as a penalization parameter to prevent the model from sacrificing too much from acc_X to increase acc_A. This is undesirable for most cases since detecting unknowns will not worth it if the classification accuracy drops dramatically. In other words, λ puts more importance on the classification accuracy than the unknown detection in this trade-off. It is calculated by simply dividing acc_X by $acc_{closed\ set}$, the accuracy of the closed set model.

The proposed method employs the InceptionTime model, state-of-the-art deep learning ensemble of five CNNs with Inception modules [17] (see [7] for an in-depth explanation) as the classifier. Since the unknown detector is independent of the classifier, InceptionTime can easily be replaced with any other classification model. This also means that the training procedure of the classifier is also separate from the unknown detector. InceptionTime is trained the same way as in [7].

Algorithm 1: Testing procedure for the OS-InceptionTime.

Input: Test sample x
Input: Classifier $f()$
Input: Barycenters for each known class $B = \{b_1, ..., b_K\}$
Input: Unknown detection thresholds τ_k^{dist} and τ_k^{cc}
Predict an initial label: $\hat{y} = f(x_r)$
`// Unknown detection part`
Calculate distances and cross-correlations **for** $k \in \{1, ..., K\}$ **do**
  Calculate the DTW distance: $d_k = DTW(x, b_k)$
  Calculate the cross-correlation: $c_k = max(correlate(x, b_k))$
end
if $d_k > \tau_k^{dist}$ ***or*** $c_k < \tau_k^{cc}$, $\forall k \in \{1, ..., K\}$ **then**
  Modify the predicted label to be the unknown class: $\hat{y} = K + 1$
end
return $\hat{y}$

Algorithm 2: Known unknowns generation algorithm.

Input: Train samples X
Input: A mean μ, and a standard deviation σ for the random noise
Define augmented data matrix $A = X$
for $i \in N$ **do**
  Generate a random noise: $noise \sim \mathcal{N}(\mu, \sigma^2)$
  Add the noise to the original sample: $A_i \mathrel{+}= noise$
end
Define splitting index $cut\_idx = L/2$
Define $temp1 = A_{1:N,1:cut\_idx}$ to store the first halves of every sample
Define $temp2 = A_{1:N,cut\_idx:L}$ to store the second halves
Switch the places of the first and second halves:
$A = concatenate(temp2, temp1)$
Reverse the order of the time steps and dimensions: $A = flip(A)$
return $\mathcal{D}_a = \{A, \overrightarrow{K+1}\}$

5 Experiments

5.1 Datasets

The 30 multivariate time series classification datasets from the UEA archive are used for all the experiments in this work. Background information about these datasets can be seen in [1]. The unknown datasets are also chosen from the archive. They are presented in Table 1 alongside the *Openness* score for each test scenario. *Openness* takes percentage values between 0% and 100%, where 0% represents a completely closed set problem. For each known dataset, two other datasets from the archive were used as the unknowns. In order to avoid cherry-picking, datasets were picked according to their sizes and shapes. The most similar ones have been used to keep the integrity as much as possible after resampling to match the shape of the original known dataset.

Table 1. The chosen unknown datasets for each train set and the openness score of each open set problem. The hybrid dataset refers to the artificially created dataset forged by concatenating PEMS-SF, InsectWingbeat, FaceDetection, FingerMovements, and HandMovementDirection along their last axes.

Training dataset	Unknown dataset 1	Openness	Unknown dataset 2	Openness
ArticularyWordRecognition	PEMS-SF	7.15%	SpokenArabicDigits	9.46%
AtrialFibrillation	HandMovementDirection	26.15%	Heartbeat	18.35%
BasicMotions	SpokenArabicDigits	35.11%	InsectWingbeat	35.11%
CharacterTrajectories	Handwriting	22.73%	PhonemeSpectra	29.29%
Cricket	SelfRegulationSCP1	5.72%	SelfRegulationSCP2	5.72%
DuckDuckGeese	Hybrid dataset*	8.71%		
EigenWorms	MotorImagery	12.29%	Cricket	34.06%
Epilepsy	CharacterTrajectories	47.48%	PhonemeSpectra	59.18%
EthanolConcentration	SelfRegulationSCP2	14.72%	Cricket	38.28%
ERing	NATOPS	20.53%	FaceDetection	10.56%
FaceDetection	InsectWingbeat	48.36%	PEMS-SF	42.26%
FingerMovements	FaceDetection	24.41%	InsectWingbeat	48.36%
HandMovementDirection	Heartbeat	14.72%	DuckDuckGeese	24.41%
Handwriting	Epilepsy	4.49%	CharacterTrajectories	15.60%
Heartbeat	PEMS-SF	42.26%	DuckDuckGeese	36.75%
JapaneseVowels	NATOPS	15.15%	FingerMovements	7.42%
Libras	NATOPS	9.95%	LSST	14.46%
LSST	FaceDetection	4.96%	SpokenArabicDigits	15.27%
InsectWingbeat	DuckDuckGeese	12.29%	PEMS-SF	15.48%
MotorImagery	DuckDuckGeese	36.75%	PEMS-SF	42.26%
NATOPS	FingerMovements	10.56%	FaceDetection	10.56%
PenDigits	LSST	19.68%	FaceDetection	6.75%
PEMS-SF	DuckDuckGeese	16.33%		
PhonemeSpectra	FaceDetection	1.87%	SpokenArabicDigits	6.38%
RacketSports	JapaneseVowels	35.11%	LSST	35.11%
SelfRegulationSCP1	SelfRegulationSCP2	24.41%	Cricket	51.49%
SelfRegulationSCP2	Heartbeat	24.41%	MotorImagery	24.41%
SpokenArabicDigits	InsectWingbeat	19.68%	FaceDetection	6.75%
StandWalkJump	MotorImagery	18.35%	Cricket	43.80%
UWaveGestureLibrary	HandMovementDirection	12.71%	PhonemeSpectra	46.55%

5.2 Experimental Results

3 baselines are considered to compare and evaluate the results of the proposed method for open set recognition. The first baseline is the most primitive one among all. It is an ensemble of small binary CNN models for each known class in the dataset (One-vs-All), with two convolutional layers followed by max pooling and two fully connected layers.

The second baseline replaces the softmax layer of the vanilla InceptionTime network with the OpenMax layer introduced in [2].

The last baseline is the class conditional VAE with the probabilistic ladder net architecture, proposed in the CGDL paper [16]. Unlike the original model, which was designed for images, 1D convolutions are used for this case.

The performance measure for the closed set classification (without the involvement of the unknown data) is the classification accuracy. The values for the performance metrics are obtained after running the algorithm three times for each testing scenario and then averaging the results. Macro F1 score, on the other hand, comes in handy when evaluating the open set performance of the models with unknown samples included in the test set, and it is the standard metric for open set papers. It will be used to evaluate the overall performance of the open set algorithms. Table 2 presents the open set performances of the algorithms for each dataset. For almost two thirds of the datasets, the proposed algortihm achieves better results than the other baselines. Detailed results for the OS-InceptionTime are given in Table 3 alongside with the optimal hyper-parameter values.

Table 2. Comparison of the open set macro F1 scores for each dataset using the unknowns from Table 1

Dataset	OvA-CNNs	OM-IT	LCVAE	OS-IT
ArticularyWordRecognition	**0.98**	0.57	0.85	0.96
AtrialFibrillation	0.19	**0.70**	0.39	0.18
BasicMotions	0.77	0.81	0.52	**0.82**
CharacterTrajectories	0.91	0.88	**0.98**	0.96
Cricket	0.83	0.75	**0.90**	0.68
DuckDuckGeese	0.33	0.25	0.35	**0.64**
EigenWorms	0.08	0.45	0.40	**0.85**
Epilepsy	0.58	0.79	0.60	**0.82**
EthanolConcentration	0.16	0.36	0.24	**0.38**
ERing	**0.90**	0.32	0.58	0.86
FaceDetection	0.40	0.44	0.15	**0.54**
FingerMovements	0.24	0.35	0.00	**0.62**
HandMovementDirection	0.20	0.13	0.30	**0.42**
Handwriting	0.18	0.16	0.20	**0.43**
Heartbeat	0.28	0.42	0.16	**0.53**
JapaneseVowels	0.87	0.90	**0.95**	**0.95**
Libras	0.60	0.66	0.74	**0.80**
LSST	0.40	**0.45**	0.08	0.36
InsectWingbeat	0.55	0.64	0.03	**0.65**
MotorImagery	0.18	0.23	0.50	**0.53**
NATOPS	0.85	0.69	**0.92**	0.89
PenDigits	0.79	0.94	0.10	**0.95**
PEMS-SF	0.67	0.76	0.55	**0.87**
PhonemeSpectra	0.10	0.34	0.13	**0.37**
RacketSports	0.51	0.63	0.67	**0.85**
SelfRegulationSCP1	0.44	0.39	**0.48**	0.46
SelfRegulationSCP2	0.27	0.19	0.30	**0.54**
SpokenArabicDigits	0.74	0.92	0.67	**0.98**
StandWalkJump	0.31	0.25	**0.45**	0.17
UWaveGestureLibrary	0.77	0.67	0.73	**0.79**
Average Results	0.50	0.53	0.46	**0.66**

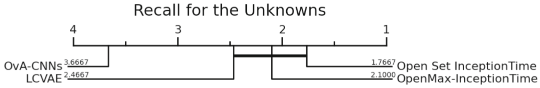
Table 3. The results for the OS-InceptionTime algorithm. Open set results are averaged for the unknown datasets given in Table 1.

Dataset	Closed set classification				Open set classification	
	InceptionTime accuracy	OS-InceptionTime accuracy	Performance decrease (%)	Hyper-parameters (α, β)	Open Set Macro F1 score	Recall for the Unknowns
ArticularyWordRecognition	0.99	0.92	−7.07	2.75, 3.25	0.96	1.00
AtrialFibrillation	0.27	0.00	−100.00	4, 0	0.18	1.00
BasicMotions	1.00	0.80	−20.00	1.75, 2.5	0.82	0.80
CharacterTrajectories	1.00	0.94	−6.00	2, 3	0.96	1.00
Cricket	0.99	0.58	−41.41	2.75, 2.75	0.68	1.00
DuckDuckGeese	0.62	0.50	−19.35	2.75, 4	0.64	1.00
EigenWorms	0.93	0.82	−11.83	2, 1.75	0.85	1.00
Epilepsy	0.97	0.79	−18.56	1.25, 1	0.82	0.81
EthanolConcentration	0.28	0.26	−7.14	1, 1	0.38	1.00
ERing	0.89	0.82	−7.87	2, 2	0.86	0.98
FaceDetection	0.67	0.39	−41.79	4, 2	0.54	1.00
FingerMovements	0.50	0.31	−38.00	4, 1.5	0.62	1.00
HandMovementDirection	0.32	0.27	−15.63	1.5, 1.5	0.42	1.00
Handwriting	0.50	0.38	−24.00	1, 1.25	0.43	1.00
Heartbeat	0.78	0.60	−23.08	1, 1.5	0.54	0.54
JapaneseVowels	0.98	0.93	−5.10	1.75, 2	0.95	1.00
Libras	0.88	0.75	−14.77	1, 1	0.8	1.00
LSST	0.45	0.45	0.00	0, 0.1	0.36	0.00
InsectWingbeat	0.71	0.67	−5.63	1, 1	0.65	0.92
MotorImagery	0.51	0.39	−23.53	1, 1.5	0.53	0.54
NATOPS	0.95	0.82	−13.68	2, 2	0.89	1.00
PenDigits	0.99	0.92	−7.07	1.5, 1.5	0.95	1.00
PEMS-SF	0.86	0.86	0.00	2.5, 3	0.87	1.00
PhonemeSpectra	0.37	0.34	−8.11	1, 1	0.37	1.00
RacketSports	0.89	0.76	−14.61	1, 1.5	0.85	1.00
SelfRegulationSCP1	0.78	0.41	−47.44	0.75, 0.5	0.46	0.61
SelfRegulationSCP2	0.51	0.37	−27.45	0.75, 1.5	0.54	0.86
SpokenArabicDigits	1.00	0.95	−5.00	1.5, 2.75	0.98	1.00
StandWalkJump	0.47	0.00	−100.00	2, 1.5	0.17	1.00
UWaveGestureLibrary	0.84	0.70	−16.67	1.5, 3	0.79	1.00

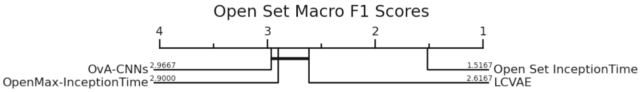
On average, the OS-InceptionTime sacrifices around 20% of the closed set classification accuracy compared to the vanilla version. In return, however, it achieves an outstanding performance for detecting the unknowns. The average recall for detecting the unknowns is 0.926. In 46 test cases out of 58 (79.3%), the proposed algorithm is able to detect all the unknowns with a perfect recall value of 1.00. In 51 cases (88%), it can detect at least half of the unknowns, and only in 7 cases (12%), it achieves 0.35 or less recall for the unknowns.

5.3 Discussion

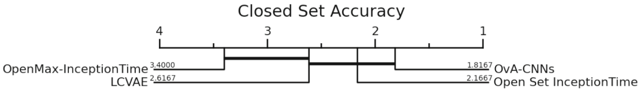
The critical difference diagrams regarding the methods used in this work are presented in Fig. 3 separately for each evaluation metric. The ranks are calculated using the Wilcoxon signed-rank test, which is used to compare repeated measurements on the same samples (in this case, test datasets). Then Holm test is used to reject the null hypothesis, i.e. the mean ranks for each pair of algorithms are not significantly different from each other. According to the Fig. 3b, the proposed Open Set InceptionTime model has the highest ranking by a significant margin, clearly separating itself from the others. However, it lacks behind the



(a)



(b)



(c)

Fig. 3. Difference diagrams of the mean ranks of the algorithms by each metric.

OvA-CNNs algorithms in terms of closed set accuracy, which is understandable because it trades-off nothing to detect unknowns.

Future Work. Since all the datasets used in this work were multivariate, the proposed method can be tested and validated on the UCR time series archive with 128 univariate TSC datasets [4]. Moreover, to trade off less closed set accuracy, better alternatives/additions to the distance and cross-correlation thresholds can be incorporated into the OS-InceptionTime, such as the difference between the forecasting errors of known and unknown samples. Finally, parallelization can be introduced to speed up the grid search for hyper-parameter optimization, as it takes the longest time to compute during the training phase with the computational complexity of $O(N^2)$.

6 Conclusion

This study presents the first ever open set model for time series classification, Open Set InceptionTime. The proposed method makes use of the class-specific barycenters of the time series to detect unknowns, and combines it with a state-of-the-art classifier. Moreover, an automated algorithm for creating the known unknown data that is required to determine the unknown detection thresholds is also presented in this work.

The experiments show that OS-InceptionTime achieves near-perfect results for unknown detection, but it trades off closed set classification accuracy while doing so. Thus, it can be considered as more suitable in situations where detecting the unknowns are more vital than the classification accuracy of the known samples. OS-InceptionTime is able to outperform all the other baselines that are adapted from computer vision to the time series classification domain. The results are validated on 30 different datasets, which proves that the proposed method is generic and applicable to various time series classification datasets.

Being the first work that develops a generic method regarding the open set recognition for time series classification, this master thesis shall act as a baseline for the future research in this field. The full implementation of the Open Set InceptionTime algorithm in Python can be found publicly on the web[1].

References

1. Bagnall, A., et al.: The UEA multivariate time series classification archive 2018. arXiv preprint arXiv:1811.00075 (2018)
2. Bendale, A., Boult, T.E.: Towards open set deep networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1563–1572 (2016)
3. Cuturi, M., Blondel, M.: Soft-DTW: a differentiable loss function for time-series. In: International Conference on Machine Learning, pp. 894–903. PMLR (2017)
4. Dau, H.A., et al.: The UCR time series archive. IEEE/CAA J. Automatica Sinica **6**(6), 1293–1305 (2019)
5. Dhamija, A.R., Günther, M., Boult, T.E.: Reducing network agnostophobia. arXiv preprint arXiv:1811.04110 (2018)
6. Ismail Fawaz, H., Forestier, G., Weber, J., Idoumghar, L., Muller, P.-A.: Deep learning for time series classification: a review. Data Min. Knowl. Disc. **33**(4), 917–963 (2019). https://doi.org/10.1007/s10618-019-00619-1
7. Fawaz, H.I., et al.: InceptionTime: finding AlexNet for time series classification. Data Min. Knowl. Disc. **34**(6), 1936–1962 (2020)
8. Ge, Z., Demyanov, S., Chen, Z., Garnavi, R.: Generative openmax for multi-class open set classification. arXiv preprint arXiv:1707.07418 (2017)
9. Jung, D.: Data-driven open-set fault classification of residual data using Bayesian filtering. IEEE Trans. Control Syst. Technol. **28**(5), 2045–2052 (2020)
10. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. Adv. Neural Inf. Process. Syst. **25**, 1097–1105 (2012)
11. Lines, J., Taylor, S., Bagnall, A.: HIVE-COTE: the hierarchical vote collective of transformation-based ensembles for time series classification. In: 2016 IEEE 16th International Conference on Data Mining (ICDM), pp. 1041–1046. IEEE (2016)
12. Oza, P., Patel, V.M.: C2ae: Class conditioned auto-encoder for open-set recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 2307–2316 (2019)
13. Petitjean, F., Ketterlin, A., Gançarski, P.: A global averaging method for dynamic time warping, with applications to clustering. Pattern Recogn. **44**(3), 678–693 (2011)
14. Schäfer, P.: The boss is concerned with time series classification in the presence of noise. Data Min. Knowl. Disc. **29**(6), 1505–1530 (2015)
15. Scheirer, W.J., de Rezende Rocha, A., Sapkota, A., Boult, T.E.: Toward open set recognition. IEEE Trans. Pattern Anal. Mach. Intell. **35**(7), 1757–1772 (2012)
16. Sun, X., Yang, Z., Zhang, C., Ling, K.V., Peng, G.: Conditional gaussian distribution learning for open set recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 13480–13489 (2020)

[1] https://github.com/tolgaakar/Open-Set-Recognition-for-Time-Series-Classification.

17. Szegedy, C., Ioffe, S., Vanhoucke, V., Alemi, A.: Inception-v4, inception-ResNet and the impact of residual connections on learning. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 31 (2017)
18. Yoshihashi, R., Shao, W., Kawakami, R., You, S., Iida, M., Naemura, T.: Classification-reconstruction learning for open-set recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4016–4025 (2019)